Motion estimation

Digital Visual Effects, Spring 2005
Yung-Yu Chuang
2005/3/23

Announcements
- Project #1 is due on next Tuesday, submission mechanism will be announced later this week.
- grading: report is important, results (good/bad), discussions on implementation, interface, features, etc.

Outline
- Motion estimation
- Lucas-Kanade algorithm
- Tracking
- Optical flow

Motion estimation
- Parametric motion (image alignment)
- Tracking
- Optical flow
Parametric motion

Tracking

Optical flow

Three assumptions
- Brightness consistency
- Spatial coherence
- Temporal persistence
Brightness consistency

Assumption
Image measurements (e.g. brightness) in a small region remain the same although their location may change.

Spatial coherence

Assumption
* Neighboring points in the scene typically belong to the same surface and hence typically have similar motions.
* Since they also project to nearby points in the image, we expect spatial coherence in image flow.

Temporal persistence

Assumption:
The image motion of a surface patch changes gradually over time.

Image registration

Goal: register a template image $J(x)$ and an input image $I(x)$, where $x = (x, y)^T$.

Image alignment: $I(x)$ and $J(x)$ are two images
Tracking: $I(x)$ is the image at time $t$. $J(x)$ is a small patch around the point $p$ in the image at $t+1$.
Optical flow: $I(x)$ and $J(x)$ are images of $t$ and $t+1$. 
Simple approach

- Minimize brightness difference

\[ E(u,v) = \sum_{x,y} (I(x+u,y+v) - J(x,y))^2 \]

Simple SSD algorithm

For each offset \((u, v)\)
compute \(E(u,v)\);
Choose \((u, v)\) which minimizes \(E(u,v)\);

Problems:
- Not efficient
- No sub-pixel accuracy

Lucas-Kanade algorithm

Newton’s method

- Root finding for \(f(x)=0\)
Taylor’s expansion:

\[ f(x_0 + \epsilon) = f(x_0) + f'(x_0) \epsilon + \frac{1}{2} f''(x_0) \epsilon^2 + \ldots \]
\[ f(x_0 + \epsilon) \approx f(x_0) + f'(x_0) \epsilon. \]

\[ e_m = \frac{f(x_m)}{f'(x_m)}. \]
\[ x_{n+1} = x_n + \frac{f(x_n)}{f'(x_n)}. \]
Lucas-Kanade algorithm

\[
E(u, v) = \sum_{x, y} (I(x+u, y+v) - J(x, y))^2
\]
\[
I(x+u, y+v) \approx I(x, y) + u I_x + v I_y
\]
\[
= \sum_{x, y} (I(x, y) - J(x, y) + u I_x + v I_y)^2
\]

\[
0 = \frac{\partial E}{\partial u} = \sum_{x, y} 2 I_x (I(x, y) - J(x, y) + u I_x + v I_y)
\]

\[
0 = \frac{\partial E}{\partial v} = \sum_{x, y} 2 I_y (I(x, y) - J(x, y) + u I_x + v I_y)
\]

\[
0 = \frac{\partial E}{\partial u} = \sum_{x, y} 2 I_x (I(x, y) - J(x, y) + u I_x + v I_y)
\]

\[
0 = \frac{\partial E}{\partial v} = \sum_{x, y} 2 I_y (I(x, y) - J(x, y) + u I_x + v I_y)
\]

\[
\begin{bmatrix}
\sum I_x^2 & \sum I_x I_y \\
\sum I_x I_y & \sum I_y^2
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix}
= \begin{bmatrix}
\sum I_x (J(x, y) - I(x, y)) \\
\sum I_y (J(x, y) - I(x, y))
\end{bmatrix}
\]

Parametric model

\[
E(u, v) = \sum_{x, y} (I(x+u, y+v) - J(x, y))^2
\]

\[
E(p) = \sum_x (I(W(x;p)) - J(x))^2
\]

translation

\[
W(x;p) = \begin{pmatrix}
x + d_x \\
y + d_y
\end{pmatrix}
\]

\[
p = (d_x, d_y)^T
\]

affine

\[
W(x;p) = Ax + d = \begin{pmatrix}
1 + d_{xx} & d_{xy} & d_x \\
d_{yx} & 1 + d_{yy} & d_y \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
x \\
y \\
1
\end{pmatrix}
\]

\[
p = (d_{xx}, d_{xy}, d_{yx}, d_{yy}, d_x, d_y)^T
\]
**Parametric model**

\[
\text{minimize} \quad \sum_x \left[ I(W(x;p + \Delta p)) - J(x) \right]^2
\]

with respect to \( \Delta p \)

\[
W(x;p + \Delta p) \approx W(x;p) + \frac{\partial W}{\partial p} \Delta p
\]

\[
I(W(x;p + \Delta p)) \approx I(W(x;p) + \frac{\partial I}{\partial x} \frac{\partial W}{\partial p} \Delta p)
\]

\[
\approx I(W(x;p)) + \frac{\partial I}{\partial x} \frac{\partial W}{\partial p} \Delta p
\]

\[\rightarrow \text{minimize} \sum_x \left( I(W(x;p)) + \nabla I \frac{\partial W}{\partial p} \Delta p - J(x) \right)^2\]

**Jacobian of the warp**

\[
\frac{\partial W}{\partial p} = \begin{pmatrix}
\frac{\partial W_1}{\partial p_1} & \frac{\partial W_1}{\partial p_2} & \cdots & \frac{\partial W_1}{\partial p_n} \\
\frac{\partial W_2}{\partial p_1} & \frac{\partial W_2}{\partial p_2} & \cdots & \frac{\partial W_2}{\partial p_n} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial W_m}{\partial p_1} & \frac{\partial W_m}{\partial p_2} & \cdots & \frac{\partial W_m}{\partial p_n}
\end{pmatrix}
\]

For example, for affine

\[
W(x;p) = \begin{pmatrix}
1 + d_{xx} & d_{xy} & d_x \\
d_{yx} & 1 + d_{yy} & d_y \\
0 & 0 & 1
\end{pmatrix}
\]

\[
\frac{\partial W}{\partial p} = \begin{pmatrix}
x \\
y \\
1
\end{pmatrix}
\]

**Parametric model**

\[
\text{minimize} \sum_x \left( I(W(x;p)) + \nabla I \frac{\partial W}{\partial p} \Delta p - J(x) \right)^2
\]

\[\rightarrow 0 = \sum_x \left[ \nabla I \frac{\partial W}{\partial p} \right]^T \left[ I(W(x;p)) + \nabla I \frac{\partial W}{\partial p} \Delta p - J(x) \right]
\]

\[
\Delta p = H^{-1} \sum_x \left[ \nabla I \frac{\partial W}{\partial p} \right]^T \left[ J(x) - I(W(x;p)) \right]
\]

**Hessian**

\[
H = \sum_x \left[ \nabla I \frac{\partial W}{\partial p} \right]^T \left[ \nabla I \frac{\partial W}{\partial p} \right]
\]
**Lucas-Kanade algorithm**

- iterate
  - warp $I$ with $W(x;p)$
  - compute error image $J(x,y) - I(W(x,p))$
  - compute gradient image
  - evaluate Jacobian $\frac{\partial W}{\partial p}$ at $(x;p)$
  - compute $\nabla I \cdot \nabla W$
  - compute Hessian
  - compute $\sum \left[ \nabla I \cdot \nabla W \right] \left[ J(x) - I(W(x,p)) \right]$
  - solve $\Delta p$
  - update $p$ by $p + \Delta p$
  - until converge

$$\Delta p = H^{-1} \sum_x \left[ \nabla I \cdot \nabla W \right]^T \left[ J(x) - I(W(x;p)) \right]$$

**Coarse-to-fine strategy**

- pyramid construction
- warp $J$ with $W$ and refine
- $\Delta \hat{a}$

**Application of image alignment**

- pyramid construction
- warp $J$ with $W$ and refine
- $\Delta \hat{a}$
- $\hat{a}_{out}$
Tracking

Optical flow constraint equation

At a single image pixel, we get a line:

\[ I_x u + I_y v = -I_t \]

\[ \frac{-I_t}{|\nabla I|} \]

“Normal flow”
**Multiple constraint**

- Assume spatial smoothness

Combine constraints to get an estimate of velocity.

---

**Area-based method**

- Assume spatial smoothness

---

**Aperture problem**

---

**Aperture problem**
Aperture problem

• Larger window reduces ambiguity, but easily violates spatial smoothness assumption

Demo for aperture problem

• http://www.sandlotscience.com/Distortions/Breathing_objects.htm
• http://www.sandlotscience.com/Ambiguous/barberpole.htm

Area-based method

• Assume spatial smoothness

\[
E(u,v) = \sum_{x,y} \left( I_x u + I_y v + I_t \right)^2
\]

\[
\frac{\partial E}{\partial u} = \sum_R (I_x u + I_y v + I_t)I_x = 0
\]

\[
\frac{\partial E}{\partial v} = \sum_R (I_x u + I_y v + I_t)I_y = 0
\]
Area-based method

The eigenvalues tell us about the local image structure.

They also tell us how well we can estimate the flow in both directions.

Link to Harris corner detector.

\[
\begin{bmatrix}
\sum R I_x^2 & \sum R I_x I_y \\
\sum R I_x I_y & \sum R I_y^2
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix}
= \begin{bmatrix}
-\sum R I_x I_y \\
-\sum R I_y I_x
\end{bmatrix}
\]

must be invertible

Textured area

Gradients in \(x\) and \(y\).

Edge

Gradients oriented in one direction.
Homogenous area

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

Weak gradients everywhere.

KLT tracking

- Select feature by $\min(\lambda_1, \lambda_2) > \lambda$
- Monitor features by measuring dissimilarity

Translational Model

What’s wrong with the translational assumption (ie constant motion within a region \( R \))?

How can we generalize it?

Affine Flow

$$E(a) = \sum_{x,y \in R} (\nabla I^T u(x; a) + I_t)^2$$

$$u(x; a) = \begin{bmatrix} u(x; a) \\ v(x; a) \end{bmatrix} = \begin{bmatrix} a_1 + a_2 x + a_3 y \\ a_4 + a_5 x + a_6 y \end{bmatrix}$$
Optimization

\[ E(a) = \sum_{x,y \in \mathbb{R}} (I_xa_1 + I_ya_2 + I_xa_3 + I_ya_4 + I_xa_5 + I_ya_6 + I_y)^2 \]

Differentiate wrt the \( a_i \) and set equal to zero.

\[
\begin{bmatrix}
\Sigma I_x^2 & \Sigma I_x I_y & \Sigma I_x & 0 & \Sigma I_y & 0 \\
\Sigma I_x I_y & \Sigma I_y^2 & 0 & \Sigma I_y & 0 & \Sigma I_x \\
\Sigma I_x & 0 & \Sigma I_x^2 & \Sigma I_x I_y & 0 & \Sigma I_y \\
\vdots & & & & & \\
0 & 0 & 0 & \Sigma I_y & 0 & \Sigma I_x \\
0 & 0 & 0 & 0 & \Sigma I_x & \Sigma I_y
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
a_4 \\
a_5 \\
a_6
\end{bmatrix}
= 
\begin{bmatrix}
-\Sigma I_x I_y \\
-\Sigma I_y \\
-\Sigma I_x I_y \\
-\Sigma I_x I_y \\
-\Sigma I_x I_y \\
-\Sigma I_x I_y
\end{bmatrix}
\]

KLT tracking

SIFT tracking (matching actually)
SIFT tracking

Frame 0  →  Frame 100

SIFT tracking

Frame 0  →  Frame 200

KLT vs SIFT tracking

- KLT has larger accumulating error; partly because our KLT implementation doesn’t have affine transformation?
- SIFT is surprisingly robust

Tracking for rotoscoping
Tracking for rotoscoping

Optical flow

Waking life

Single-motion assumption

Violated by
- Motion discontinuity
- Shadows
- Transparency
- Specular reflection
- ...

Waking Life
What is the “best” fitting translational motion?

Simple problem: fit a line

\[ E(a,b) = \sum (y_i - (ax_i + b))^2 \]
Least-square fit

Robust statistics
- Recover the best fit for the *majority* of the data
- Detect and reject outliers

Approach

Robust weighting

Influence is proportional to the derivative of the $\rho$ function.

Tukey’s biweight.

Want to give less influence to points beyond some value.

Beyond a point, the influence begins to decrease.

Beyond where the second derivative is zero – outlier points
Robust estimation

\[ E(a) = \sum_{x,y \in R} \rho(I_x u + I_y v + I_t, \sigma) \]

Minimize: differentiate and set equal to zero:

\[ \frac{\partial E}{\partial u} = \sum_{x,y \in R} \psi(I_x u + I_y v + I_t, \sigma) I_x = 0 \]
\[ \frac{\partial E}{\partial v} = \sum_{x,y \in R} \psi(I_x u + I_y v + I_t, \sigma) I_y = 0 \]

*No closed form solution!*

---

Fragmented Occlusion

---

Results

---

Results
Regularization and dense optical flow

Assumption
* Neighboring points in the scene typically belong to the same surface and hence typically have similar motions.
* Since they also project to nearby points in the image, we expect spatial coherence in image flow.

Formalize this Idea

Noisy 1D signal:

Noisy measurements \( u(x) \)

Regularization

Find the “best fitting” smoothed function \( v(x) \)

Noisy measurements \( u(x) \)

Minimize: Faithful to the data Spatial smoothness assumption

\[
E(v) = \sum_{x=1}^{N} (v(x) - u(x))^2 + \lambda \sum_{x=1}^{N-1} (v(x+1) - v(x))^2
\]
Discontinuities

What about this discontinuity? What is happening here? What can we do?

Robust Regularization

Minimize:

\[ E(v) = \sum_{x=1}^{N} \rho(v(x) - u(x), \sigma_1) + \lambda \sum_{x=1}^{N-1} \rho(v(x+1) - v(x), \sigma_2) \]

Treat large spatial derivatives as outliers.

Optical flow

Outlier with respect to neighbors.

Robust formulation of spatial coherence term

\[ E_S(u,v) = \rho(u_x) + \rho(u_y) + \rho(v_x) + \rho(v_y) \]

“Dense” Optical Flow

\[ E_D(u(x)) = \rho(I_x(x)u(x) + I_y(x)v(x) + I_t(x), \sigma_D) \]

\[ E_S(u,v) = \sum_{y \in G(x)} [\rho(u(x) - u(y), \sigma_S) + \rho(v(x) - v(y), \sigma_S)] \]

Objective function:

\[ E(u) = \sum_x E_D(u(x)) + \lambda E_S(u(x)) \]

When \( \rho \) is quadratic = “Horn and Schunck”
Example

Quadratic:

Robust:

Applications of Optical Flow

Magnitude of horizontal flow

Impressionist effect. Transfer motion of real world to a painting
Smooth gradient

Textured brush

Edge clipping

Temporal artifacts

Frame-by-frame application of the NPR algorithm
Temporal coherence

What dreams may come

Reference

• J. Shi and C. Tomasi, Good Features to Track, CVPR 1994, pp593-600.
• Peter Litwinowicz, Processing Images and Video for An Impressionist Effects, SIGGRAPH 1997.
• Aseem Agarwala, Aaron Hertzman, David Salesin and Steven Seitz, Keyframe-Based Tracking for Rotoscoping and Animation, SIGGRAPH 2004, pp384-591.